Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 60 (2015) 1061 – 1069

Procedia Computer Science

19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems

Solving bin packing problem with a hybrid genetic algorithm for VM placement in cloud

Mohamed Amine KAAOUACHE[a,]*, Sadok BOUAMAMA[a]

[a] *COSMOS Lab, National School of Computer Science, University Campus Manouba, Manouba 2010, Tunisia*

Abstract

The Bin Packing Problem's purpose (BPP) is to find the minimum number of bins needed to pack a given set of objects of known sizes so that they do not exceed the capacity of each bin. This problem is known to be NP- Hard. In this paper, we propose an hybrid genetic algorithm using BFD (Best Fit Decreasing) to deal with infeasible solution due to the bin-used representation. Experimental results showed the effectiveness of our approach for infeasible chromosomes thereby improving the quality of the obtained solution. This will give a good result for the virtual machine placement in Cloud to minimize energy since it looks like a BPP.

Keywords: Genetic algorithms, Bin Packing, Heuristics, Infeasible solutions, Virtual machine placement, Cloud Computing

1. Introduction

The one-dimensional Bin Packing Problem (BPP) is defined as follows. Given an unlimited positive integer number of bins with a fixed capacity C, and a set of n items, each with a specific weight $0< w_i< c$, is it possible to 'pack' all the items into N bins without exceeding the capacity of any bin. There exist a partition of the set n of items into m subsets, each of which corresponds to the content of one bin [1]. BPP is applicable in several area such as logistic and industrial applications. The term "bin" here is in fact a generic name which could stand for a "container", as in the "transportation" context, "work stations " in industrial assembly lines (line balancing), "a space in time" in scheduling, or a "surface area", as in metal working, for example. This NP-complete decision problem naturally gives rise to the associated NP-hard optimization problem, which demands a large amount of resources for its solutions.

In Cloud, improving the system performance and optimizing the organizational resources has caught the interest of researchers and users of all computing systems. This management leads to minimizing energy through virtual machine placement and translates into saving a significant amount of money and natural resources.

In this paper we concentrate only on the one dimensional bin packing problem. We first present some solution procedures and representation schemes so far suggested for the problem. Secondly we present our hybrid genetic algorithm utilizing BFD to deal

[*] Corresponding author. Tel.: +966 530122156 .
E-mail address: kaaouache.amine@gmail.com

doi:10.1016/j.procs.2015.08.151

with infeasible solution. Then we present an adaptation of our approach for virtual machine placement. Finally, some computational results are given in section 5, followed by summary and conclusions.

2. Bin Packing Formulation Problem

2.1. Bin packing Research

BPP is solved with various methods proposed by many researchers. In fact, it is known that the bin packing problem is NP-Hard. Hence, heuristic procedures such as next fit (NFD), first-fit decreasing method (FFD) and best-fit decreasing method (BFD), are proposed for this problem. In these methods, items are first sorted in the order of non-increasing size, and then first-fit (E) and best-fit (BF) procedures are applied for the items in turn, respectively [7].

Since these methods are easily implemented, they are usually embedded in a genetic algorithm to enhance its performance. Often, they are embedded in genetic algorithm approach to enhance the genetic search to find a better nearly optima solution for the BPP problem. The algorithms using this dynamic characteristic are usually referred to as on-line algorithms. In addition their optimal values provide upper bounds, and through proven inequalities lower bounds, for the problem.

The packing used by first fit or best fit uses no more than 17/10 OPT+2 bins (where OPT is the optimal number of bins in an optimal solution), while first fit decreasing or best fit decreasing uses at most 11/9 OPT+4 bins. A better bound for first fit decreasing, given in [12] is 11/9 OPT+1 [4]. The time complexity of both FFD and BFD is O(n log n). BPP is also dealt with meta-heuristics such as Particle Swarm, Ant Colony and Genetic Algorithm. The last one has been the subject of several improvements that have proved their effectiveness in bin packing problem.

2.2. Bin Packing representation scheme

To resolve the bin packing problem three representation schemes have been proposed, bin based representation, object-based representation, and group-based representation [5].

2.2.1. Bin-Based representation

In bin-based representation, a chromosome has a fixed length equal to the number of objects. Each bin is represented by a gene located in a chromosome that indicates the object number placed in the bin represented by that gene. For example, the chromosome 4 3 2 5 1 indicates that object 1 is placed in bin 4, object 2 is placed in bin 3, object 3 is placed in bin 2, and so on, (Table 1).

Table 1. Bin-based representation for the example

Objects	1	2	3	4	5
Bins	4	3	2	5	1

The fixed length of each chromosome present an advantage which makes it suitable for application of genetic operators. A drawback of this method, as mentioned by Falkenauer [2], is that it creates many redundant solutions, which exponentially increases with the number of bins, which decreases in turn the efficiency of the GA. In addition, it might create infeasible chromosome solutions which exceed the bin's capacity.

2.2.2. Object-Based representation

In the object-based representation, a chromosome represents a permutation of objects. Then the chromosome is partitioned based on number of objects placed in bins. For example, if we have five objects, then a permutation of the numbers 1 to 5 is 1 2 3 4 5. A partition of this chromosome could be 1 2 | 3 4| 5, which indicates that objects 1 and 2 are placed in bin 1, objects 3, 4 are placed in bin 2, and object 5 is placed in bin 3. This representation has some disadvantages too. For example, the following chromosomes with the indicated distributions represent the same solution as above:
2 1 | 4 5| 3, 3| 4 5 | 2 1 , 3 | 2 1 | 4 5, etc.

2.2.3. Group-Based representation

This representation proposed by Falkenauer [1], [2] is in fact group oriented, and represents the most adequate and best suited to the representation of the problem. It is more important to know how the objects are grouped together as to how they are placed. This method involves two phases, the first looks like the bin-based representation [4].

The second phase represents an encoding of the bins used in a one-to-one correspondence to genes. The following example explains this encoding, the chromosome 1 4 6 5 6 2 means that object 1 is placed in bin 1, object 4 in bin 5, objects 3 and 5 in bin 3, object 4 in bin 5 and object 6 in bin 2. Using letters to represent the bins, then this chromosome can be represented as ADFEFB, where A = {1}, D = {2}, F = {3, 5}, E = {6}, and B = {4} (Table 2).

Table 2. Group-based representation for the example

Objects	1	2	3,5	6	4
Bins	A	D	F	E	B

This representation gives an important advantage. The genes represent both objects and groups. The group part is actually used in genetic operations, while the object part is used to identify the elements of a group. The disadvantage, nevertheless, is that the chromosomes have a variable length.

2.3. Mathematic Model

The one dimensional bin packing problem can be formulated as [4]:

$$min(z) = \sum_{j=1}^{n} y_j \quad (1)$$

$$\sum_{i=1}^{n} w_i x_{ij} \leq C y_j \ , \forall j \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad , \forall j \quad (3)$$

$$x_{ij}, y_j = 0,1 \qquad , \forall\, i, j \quad (4)$$

Where,

$$y_j = \begin{cases} 1 & if\ bin\ j\ is\ used \\ 0 & otherwise \end{cases} \quad (5)$$

$$x_{ij} = \begin{cases} 1 & if\ object\ i\ is\ placed\ in\ bin\ j \\ 0 & otherwise \end{cases} \quad (6)$$

The objective minimizes the total number of bins used. Constraints (2) ensure that the weights of objects placed in bin j, do not exceed the capacity of the bin. We don't forget that all the bins have the same capacity C, but in general this need not be the case. Finally, constraints (3) ensure that each object is placed only in one bin.

Since the bin packing problem is NP-Hard. Hence, heuristic procedures such as next fit, first fit, first fit decreasing, best fit, best fit decreasing, to name a few, are proposed for this problem. Thanks to their ease of implementation, these procedures are usually embedded in a genetic algorithm to enhance its performance. We have adopted this practice here, too. In addition their optimal values provide upper bounds, and through proven inequalities lower bounds, for the problem.

3. Hybrid Genetic Algorithm for BP

3.1. Genetic Algorithm Researches

GA is a search heuristic originally proposed by Holland [8] that mimics the process of natural selection. Since then, this heuristic also called a meta-heuristic has evolved into a powerful method for solving many hard combinatorial optimization problems and generating useful solutions to search problems [1]. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions using techniques inspired by natural evolution, such as mutation, selection, and crossover [4]. With these three simple operators a Genetic Algorithm works as follows.

```
Generate an initial population
While non stopping criteria is met, do
    • Choose pairs for breeding
    • Perform cross-over to generate off-springs
    • Evaluate the fitness of new off-springs
    • Generate a new population
end while
```

The different evolved version of genetic algorithm have proven effective in solving NP complete problem including the BPP, among these versions, the hybrid genetic algorithms which lead to good results. The most relevant results have been obtained by means of hybrid and heuristic algorithms.

Martello and Toth [9] proposed a branch-and-bound procedure (MTP) and a dominance criterion to reduce the search space. Falkenauer [2] proposed a hybrid grouping genetic algorithm (HGGA) that uses a special encoding scheme for groups of items and a local optimization heuristic inspired by the dominance criterion of Martello and Toth. Scholl et al. [10] developed a hybrid technique (BISON) that combines a tabu search-based heuristic with a branch-and-bound method, using a "dual" strategy that comprises minimizing the fullness of the bins given a fixed number of bins. Schwerin and Wäscher [11] proposed a new lower bound for BPP that is based on the cutting stock problem, and they integrated it into the MTP procedure of Martello and Toth, obtaining high-quality results with their new procedure (MTPCS). Alvim et al. [13] presented a hybrid improvement heuristic (HI_BP) in which the dual strategy used by Scholl et al. [12] is reintroduced, in addition to the search space reduction techniques of Martello and Toth [9] and the use of lower bounding strategies.

This procedure uses a load redistribution method that is based on dominance, differencing, and unbalancing and the inclusion of an improvement process that utilizes a tabu-search, obtaining the best reported results until that moment. The study of the literature that is related to the BPP solution revealed that, until now, the best algorithms are the following: Perturbation-SAWMBS [5], HGGA [2] and HI_BP [7].

3.2. Hybrid Genetic algorithm approach HGBF_BP

3.2.1. Basic features of HGBF_BF

In GA model, a standard encoding scheme is using binary encoding or real number encoding to generate the population of variables, with different crossover and mutations. In HGBF_BP, a special encoding scheme will be used in order to illustrate special encoding structures of BP problems to form chromosomes. In the standard encoding scheme, it usually generates the population by randomly allocating numbers in chromosome to maintain the randomness of initial population which is suitable for most kinds of production problems. However, random allocation of genes into chromosome in BP problem will end up with some infeasible chromosomes in the population. In this paper, the proposed encoding scheme for the HGBF_BP is a bin-oriented scheme with new approach to deal with infeasible chromosomes. The chromosome i refers to the same item each time. Items are in constant order so that it never need modification. We are moving bins instead of moving items.

Selection operator selects to be mate chromosomes according to their fitness values. Several number of selection methods can be applied to GA including random selection, roulette wheel selection, stochastic universal sampling and elitism, etc. Roulette wheel selection operator is chosen as the selection method for the proposed HGBF_BP. First, a fitness value is calculated for each chromosome in population. The fitness value of a chromosome is divided by the total fitness value and obtained its relative fitness value. Then two chromosomes are chosen to form the next generation according to their relative fitness value.

In the crossover process, it demonstrates the direction on how to generate new offspring naturally, based on the concept of biological interchange. Two-point crossover is chosen for the interchange between selected pair of chromosomes in order to increase the chance of producing offspring with higher fitness values.

Mutation operation is a process to avoid reaching local optimum instead of global optimum in GA. Beside the crossover operator, mutation operator will allow chromosomes randomly remove one or more gene in its own chromosome. This operation will be terminated after the selected chromosomes have been rearranged. The entire processes of HGBF_BP including calculation of fitness values, selection, crossover, and mutation, they will be repeated until the number of generations has reached the predefined value. The final solution of HGBF_BP present the objective function which is the number of bins being used and cost of bins as stated at the model.

3.2.2. Hybrid Genetic algorithm approach to solve infeasible solution HGBF_BP

We adopt in HGBF_BP a bin-based representation solution for BPP. The advantage of this representation is that it provides a fixed length of each chromosome which makes it suitable for application of genetic operators. As mentioned by Falkenauer [2], this representation has drawbacks. The problem of redundancy in the coding still arises, which exponentially increases with the number of bins, which decreases in turn the efficiency of the GA. In spite of these potential problems, existing work [14] did not suggest that such problems had been observed in practice. Further, a search of the literature showed that a similar approach had been tried for clustering and partitioning problems [15], with reasonable success.

The other disadvantage of bin-based representation is infeasible chromosome solutions which exceed the bin's capacity. To solve problem, we proposed a hybrid approach based on BFD to deal with infeasible chromosomes. We opt for this choice since GA+BFD appears superior to GA+FFD, which would seem to indicate that it is worth hybridizing with as good a heuristic as possible. By way of comparison, Falkenauer and Delchambre [2] reported results which appear qualitatively to be very similar to GA+BFD, although the actual problem instances they used were of course different [6].

In our approach HGBF_BP, we try to honor the initial packing as much as possible. In case of infeasible chromosomes which exceed bin's capacity, we proceed with BFD packing strategy to place that package and repair the chromosome. First, we pack the items and in case of exceeding capacity, we save all those packages who's original bins were full. We, then apply the BFD algorithm on all the rest who's bins were full up. If after this procedure, there remain the item that you cannot pack and these bins are overloaded, then we dump out them. And if after a number of tries we cannot pack these items, we mark this chromosome as unsalvageable because it is impossible to fit the items in the bins. The repaired chromosome will be the one used to get a fitness value.

Algorithm 1. Hybrid Genetic Best fit (HGBF_BP)

```
b : bin's size
bl : bin's length
bc : bin's content
c : chromosome content
ci : chromosome index
cl : cannotPackList length
tr : number of tries
bbi : best bin index

for i:= 0 to cl do
    if c < bl and ci + bc < b then
    bc:=bc + c  // put item into bin
    else cannotPackList(i):=bc // save all items whose original bins were full
    end if
end for
for r:= 0 to cl do
    while bbi < 0 and tr < bl do
        BFD (c)
        if bbi < 0 then
            cannotPackList(r):=ci
            tr++
        end if
    end while
    if tr > bl then return null  // it is impossible to fit the items in the bins
    end if
    bc:=bc + c
    ci:=bbi
end for
```

3.3. Fitness Function

Falkenauer and Delchambre [2] has introduced The fitness evaluation made by means of the cost function. Eq. 1 defines the cost function for the BPP, where m is the number of bins that are used in the solution, Ii is the sum of the item weights in bin i, and c is the capacity of the bins and k is a heuristic exponential factor.

$$F_{BPP} = \frac{\sum_{i=1}^{m} (Ii/c)^{k}}{m}, \; k = 2. \tag{7}$$

The cost function evaluates the average of the squaring of the 'bin efficiency', which measures the exploitation of a bin's capacity. After several experiments of the value of k which expresses a concentration on the almost full bins in comparison to less filled ones, the best value of k was 2. However, according to Stawowy [3], k = 4 gives slightly better results. The objective of our algorithm is to maximize the fitness values of the individuals in the population [5].

4. Adapting our approach for virtual machine placement in cloud

4.1. Virtual machine placement Problem

Virtual machine placement is the process of mapping virtual machines to physical machines, i.e. virtual machine placement is the process of selecting the most suitable host for the virtual machine. The process involves classifying the virtual machines hardware and resources requirements and the anticipated usage of resources and the placement goal. The placement goal can either be maximizing the usage of available resources or it can be saving of power by being able to shut down some servers or keep them in standby state. Placement algorithms can be broadly classified into two categories on the basis of their placement goal, one of them is the Power based approach. The other is Application QoS based approach [17].

4.1.1. Power Based Approach

The necessity of power management has become increasingly evident in computing environments. This necessity is driven by two factors:

- The increasing demands on power by both computing and cooling resources in a data center.
- The rising cost of power.

The main aim of these approaches is to map virtual machines to physical machines in such a way, so that the servers can be utilized to their maximum efficiency, and the other servers can be either hibernated or shut down depending on load conditions.

4.1.2. Application QoS Approach

These algorithms manage the mapping of virtual machines onto physical hosts with the aim of maximizing the quality of service (QOS) delivered. By continuously monitoring virtual machine activity and employing advanced policies for dynamic workload placement, such algorithms can lead to better utilization of resources and less frequent overload situations eventually leading to savings in cost.

4.2. Contribution of HGBF_BP in solving VM placement

Since the VM placement problem can be designed as a bin packing problem by considering Physical machines as bins and the Virtual machines to be placed can be considered as objects to be filled in the bin, it is estimated that the motivation behind our algorithm HGBF_BP is to reduce the costs involved. Since we opt for the power based approach, the costs can be reduced by optimally using the resources available like CPU, memory etc., and by shutting down the servers that have no load or hibernate them.

This will automatically generate the reduction of power required for non-computational resources like cooling, lighting, etc. [18]. The aim would be to deliver a solution that is nearly optimal in terms of the number of hosts used and the efficiency of packing of the hosts.

5. Results

We report in this section the computational experiments performed with the hybrid improvement heuristic on a broad set of test problems. Heuristic HBF_ BP was coded in JAVA, All experiments were performed on a Intel ® Core™ 2 Duo 2.26 GHz Pentium IV with 3.00 GB of RAM memory usable. The results for a few sample problems with 4, 9, 10, 20, 40, 50 and 120 objects are given in Table 3, below. Note that in all problems listed HGBF_BP achieves the optimal solution except the 120 objects' result which is too close to the optimal value with a difference of a bin.

This good result is partially due to the fact that we have corrected infeasible solutions to prevent overflow of the bin. This improvement will reduce the computation time but at the cost of reducing the accuracy of the solution. In our case, the time needed to solve the problem with 120 objects was less than a second, which proves the effectiveness of our approach. The fact that we minimize the number of bins, it will lead to minimizing the consumption of energy in the data center.

Table 3. The results for some test problems from [16]

Set	The number of objects	Capacity of bins	FF	HGBF_BP	Optimum
1	120	150	63	50	49
2	50	100	25	23	23
3	40	70	23	19	19
4	20	45	11	10	10
5	10	20	6	6	6
6	9	14	7	6	6
7	4	6	3	2	2

We tried also another example where the bin size is set to 100, number of items is 400 and their sizes are chosen randomly between 20 and 100. It shows progress of chromosomes' fitness values and number of bins used. X-axis represents generations of the population and Y-axis represents fitness value or number of bins used by the solution.

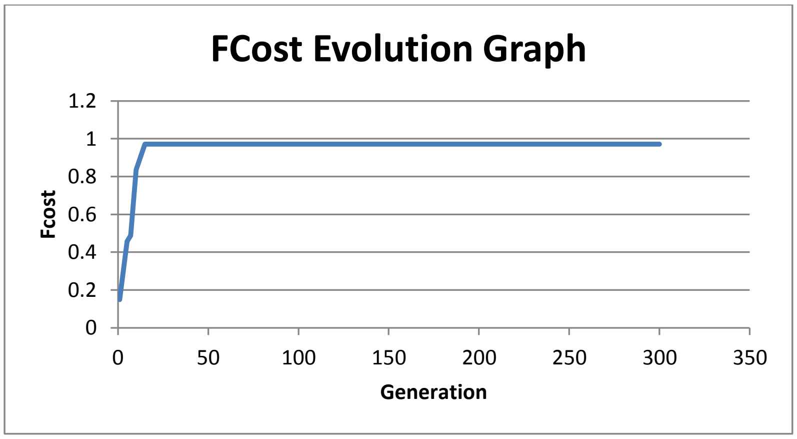

Fig. 1. Cost Function evolution according to generations.

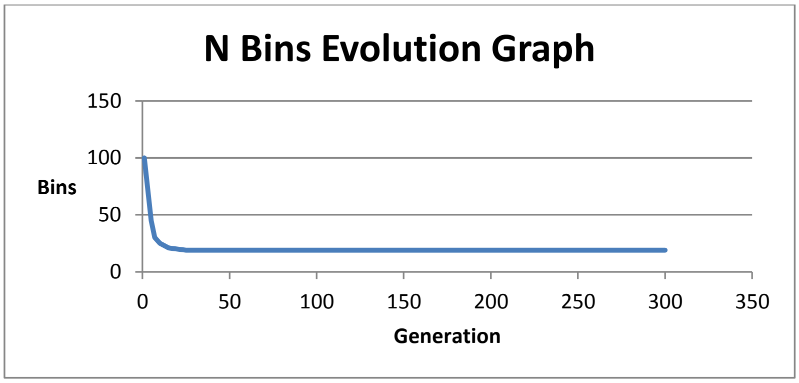

Fig. 2. Evolution of the number of bins according to generations.

By analyzing Fig. 1 and Fig. 2, the cost function rapidly converges to a nearly constant value. Therefore, the number of bins decreases toward its optimal value. This proves the effectiveness of our approach to minimize energy through virtual machine placement in cloud.

6. Summary and conclusions

In this paper, we considered the one dimensional bin packing problem. We briefly reviewed some heuristic procedures and representation schemes suggested for the problem. We then proposed a new approach based on correcting infeasible chromosomes to prevent overflow of the bin. Hence, the new proposed chromosomes was suitable for the application of the genetic operators and contributed to reduce the execution time. To test the efficiency of the proposed solution and the algorithm, we tested few sample problems. Our limited computational results attests the efficiency of the proposed approach.

In addition, since our approach has optimized the BPP, we can deduce that we can minimize the energy emitted by the data center in Cloud. This optimization is done through VM placement that leads to use a minimum number of physical machines and the other servers can be either hibernated or shut down.

Acknowledgements

The authors express their deepest gratitude to the anonymous reviewers for their invaluable comments and suggestions which helped greatly in improving the quality of the paper.

Appendix A. Object problem's weights

A.1. Weights for the 120 object problem

97; 57; 81; 62; 75; 81; 23; 43; 50; 38; 60; 58; 70; 88; 36; 90; 37; 45; 45; 39; 44; 53; 70; 24; 82; 81; 47; 97; 35; 65; 74; 68; 49; 55; 52; 94; 95; 29; 99; 20; 22; 25; 49; 46; 98; 59; 98; 60; 23; 72; 33; 98; 80; 95; 78; 57; 67; 53; 47; 53; 36; 38; 92; 30; 80; 32; 97; 39; 80; 72; 55; 41; 60; 67; 53; 65; 95; 20; 66; 78; 98; 47; 100; 85; 53; 53; 67; 27; 22; 61; 43; 52; 76;64; 61; 29; 30; 46; 79; 66; 27; 79; 98; 90; 22; 75; 57; 67; 36; 70; 99; 48; 43; 45; 71; 100; 88; 48; 27; 39

A.2. Weights for the 50 object problem

12; 45; 90; 92; 30; 40; 42; 51; 17; 18; 12; 25; 32; 47; 41; 63; 60; 51; 81; 73; 51; 40; 15; 20; 11; 12; 9; 13; 91; 92; 75; 44; 71; 32; 35; 16; 29; 44; 39; 58; 16; 23; 9; 91; 26; 42; 84; 98; 50; 62

A.3. Weights for the 40 object problem

12; 13; 11; 40; 22; 60; 61; 63; 11; 10; 19; 31; 32; 37; 25; 14; 21; 38; 51; 59; 40; 45; 54; 62; 59; 40; 13; 31; 17; 20; 26; 36; 15; 12; 9; 10; 27; 31; 55; 40

A.3. Weights for the 20 object problem

17; 19; 12; 11; 17; 18; 17; 4; 5; 21; 10; 23; 37; 32; 29; 40; 41; 30; 21; 11

A.4. Weights for the 10 object problem

14; 15; 12; 2; 4; 8; 13; 19; 20; 7

A.4. Weights for the 9 object problem

5; 7; 3; 5; 12; 11; 10; 11; 9

A.5. Weights for the 4 object problem

2; 2; 4; 4

References

1. Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2(1), 5–30.
2. Falkenauer E, Delchambre A. A genetic algorithm for bin packing and line balancing. *Proceedings of the IEEE 1992 international conference on robotics and automation (RA92)*, May 10–15, Nice, France, 1992.
3. Stawowy, A. Evolutionary based heuristic for bin packing problem. *Computers & Industrial Engineering*. 2008, 55, 465–474.
4. Mitsuo Gen, Runwei Cheng. *Genetic algorithms and engineering optimization* . New York [u.a.] : John Wiley & Sons, 2000.
5. Tansel Dokeroglu , Ahmet Cosar. Optimization of one-dimensional Bin Packing Problem with island parallel grouping genetic algorithms. *Computers & Industrial Engineering* 75 (2014) 176–186.
6. C. Reeves. Hybrid genetic algorithms for bin-packing and related problems. *Annals of Operations Research* 63(1996)371 - 396
7. Iima H, Yakawa T. A New Design of Genetic Algorithm for Bin Packing. *Evolutionary Computation*, 2003. CEC '03.
8. Holland J. Adaptation in natural and artificial systems, *University of Michigan press. Ann Arbor, MI, MIT press, Cambridge, MA*, (1975,1992).
9. Martello S, Toth P. *Knapsack problems: algorithms and computer implementations*. New York: John Wiley & Sons, 1990.
10. Scholl A, Klein R, Jürgens C. BISON: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem. *Computers and Operators Research*. 24 (1997), 627–645.
11. Schwerin P, Wäscher G. A new lower bound for the bin-packing problem and its integration to MTP. *Pesquisa Operational* 1999. 19:111–29.
12. Yue M. A simple proof of the inequality FFD(L)≤ (11/9)OPT(L) + 1, for all L, for the FFD bin-packing algorithm. *Acta. Math. Appl. Sinica*. (7), 321-331,1991.
13. Alvim ACF, Ribeiro CC, Glover F, Aloise DJ. A hybrid improvement heuristic for the one-dimensional bin packing problem. *Journal Of Heuristics*.10(2):205–229.2004.
14. D. Smith, Bin packing with adaptive search, *Int. Conf. on Genetic Algorithms and their Applications (Lawrence Erlbaum, Hillsdale, NJ,* 1985).
15. D.R. Jones and M,A. Beltramo, Solving partitioning problems with genetic algorithms, *4th Int. Conf. on Genetic Algorithms (Morgan Kaufmann, San Mateo, CA*, 1991).
16. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html
17. Pathan N, Shetty B. Virtual Machine Placement in Cloud. *IJCSIT* Vol. 5 (2), 1833-1835. 2014.
18. Madhusudhan M, Chandra Sekaran K. A genetic algorithm approach for virtual machine placement in cloud. *ERCICA* 2013.